LLM-Codec: Neural Audio Codec Meets Language Model Objectives

Ho-Lam Chung[*,‡] **Yiming Chen**[‡*] **Hung-yi Lee**[†]
[*]Graduate Institute of Communication Engineering, National Taiwan University
[†]NTU Artificial Intelligence Center of Research Excellence (NTU AI-CoRE)
[‡]ASUS Intelligent Cloud Services

Abstract

Neural audio codecs are widely used as tokenizers for spoken language models, but they are optimized for waveform reconstruction rather than autoregressive prediction. This mismatch injects acoustically driven uncertainty into the discrete token space and increases language-model perplexity. We propose LLM-CODEC, which augments codec training with language-model-facing objectives while keeping both codec and LLM architectures unchanged. LLM-CODEC introduces (i) future token prediction with Medusa-style multi-step heads to encourage multi-step predictability, and (ii) semantic alignment that matches audio and text representations via a memory-bank contrastive loss. A differentiable Gumbel bridge enables end-to-end gradients from these objectives to the codec encoder. On SALMon speech coherence, token LMs trained on LLM-CODEC reach 61.6% accuracy (+12.1 points over AUV) while reducing perplexity 35×. On Codec-SUPERB-tiny, LLM-CODEC improves speech Mel distance by 5.0% over AUV while simultaneously achieving the learnability gains, demonstrating that reconstruction fidelity and token predictability can be improved together. [1]

1 Introduction

Following the success of large language models (LLMs), spoken language models (SLMs) have emerged as a promising paradigm for speech generation. SLMs represent speech as discrete token sequences and model them with autoregressive LLM backbones. Most SLMs adopt neural audio codecs, such as EnCodec (Défossez et al., 2022) and SoundStream (Zeghidour et al., 2021), which provide a bidirectional mapping between waveforms and discrete speech tokens. This design provides a unified interface for modeling speech and text within the same vocabulary space.

However, a fundamental tension exists between how codecs and LLMs are trained. Codecs are optimized for *reconstruction*, i.e., minimizing distortion between the waveform x and its reconstruction $\hat{x}$ (e.g., $\|x - \hat{x}\|$), whereas LLMs are optimized for *prediction*, i.e., maximizing next-token likelihood. These objectives favor different representations. To achieve high-fidelity reconstruction, a codec must preserve fine-grained acoustic factors such as pitch micro-variations, phase, breathing, and background conditions. Many of these factors are weakly tied to linguistic content. In a discrete token space, these factors behave like stochastic variations. This stochasticity increases token entropy and makes the resulting sequences harder for an LLM to model. In generation, the mismatch can manifest as repetition, semantic drift, and hallucinated content.

This paper asks a simple question. Can we retrain an existing codec encoder so that its discrete tokens remain reconstructable, but become predictable under autoregressive language modeling?

These observations suggest a simple principle: if an LLM must predict speech tokens, then the codec should be trained to emit tokens that are predictable under language modeling while preserving linguistic content. Motivated by this, we propose LLM-CODEC, which augments codec training with two LLM-facing regularizers.

First, we introduce **Future Token Prediction (FTP)**, which attaches K auxiliary heads to predict multiple future tokens, implemented with a Medusa-style (Cai et al., 2024) design and inverse-distance weighting. Unlike standard next-token prediction, which models only one-step dependencies, FTP encourages longer-range structure by capturing linguistic units (e.g., phonemes and words) that often span multiple tokens.

Second, **Semantic Alignment (SA)** addresses the risk of producing predictable but semantically

[*]Corresponding author.
[1]Code & Model will be released at https://github.com/voidful/llm-codec

Findings of the Association for Computational Linguistics: ACL 2026, pages 26251–26265

arbitrary codes by aligning speech and text representations within the LLM using layer-wise cosine alignment and a memory-bank contrastive objective.

Finally, to enable end-to-end optimization through vector quantization, we further introduce a differentiable Gumbel-Softmax bridge that preserves discrete tokens in the forward pass while providing smooth gradients in the backward pass.

We conduct two complementary evaluations on speech coherence and audio reconstruction benchmarks. On SALMon speech coherence benchmark, LLM-CODEC produces tokens that are more amenable to language modeling: LMs trained on LLM-CODEC tokens reach 61.6% overall accuracy, substantially exceeding all baselines (48–50%). Moreover, LLM-CODEC reduces token-level perplexity by 35× compared to the base codec, confirming that objectives, not model capacity, determine token predictability. On Codec-SUPERB-tiny (Wu et al., 2024), which measures codec reconstruction quality, LLM-CODEC improves speech Mel distance by 5.0% over AUV while maintaining competitive perceptual quality. This result demonstrates that improved token learnability does not require sacrificing signal fidelity. Finally, LLM-CODEC is simple to adopt, as it modifies only the training objectives without changing model architectures.

Overall, our major contributions include:

- **Objective mismatch.** We formalize the mismatch between reconstruction-trained codecs and prediction-trained LMs, and we show that it inflates audio-token uncertainty and LM perplexity.
- **LLM-CODEC Training Framework.** We propose LLM-CODEC training framework, which utilizes FTP and SA as complementary regularizers to address predictability and semantics respectively. We use a hard Gumbel-Softmax bridge to backpropagate through quantization while keeping discrete tokens in the forward pass.
- **Comprehensive Evaluation.** On SALMon, LLM-CODEC improves speech-coherence accuracy to 61.6% (+12.1 over AUV). Token-level perplexity drops 35×. On Codec-SUPERB-tiny, speech Mel distance improves by 5.0%, and ablation confirms that reconstruction and learnability gains are additive.

2 Related Work

Neural audio codecs. SoundStream (Zeghidour et al., 2021) and EnCodec (Défossez et al., 2022) established neural audio compression with vector quantization and adversarial training. BigCodec (Xin et al., 2024) explores a large single codebook. WavTokenizer (Ji et al., 2025) improves codebook utilization and targets better language modeling compatibility. These methods optimize reconstruction quality. LLM-CODEC modifies the training objective to incorporate language-model-facing signals.

Semantic speech tokens and self-supervised learning. Self-supervised models such as wav2vec 2.0 (Baevski et al., 2020) and HuBERT (Hsu et al., 2021) learn discrete or pseudo-discrete units that correlate with phonetic content. SpeechTokenizer (Zhang et al., 2024) and related lines separate semantic and acoustic factors for speech LLMs. These approaches often introduce additional encoders or decoders. LLM-CODEC keeps the codec architecture unchanged and instead reshapes the codec via LLM-aligned losses.

Multi-step prediction. Medusa (Cai et al., 2024) introduces multiple decoding heads to predict several future tokens for faster LLM inference. FlowSLM (Chou et al., 2025) argues that spoken language modeling benefits from constraints beyond one-step prediction. LLM-CODEC adapts the Medusa idea as a training-time regularizer to encourage multi-step predictability in codec tokens.

Audio-text alignment. CLAP (Elizalde et al., 2023) aligns audio and text for retrieval in a shared embedding space. SpeechGPT (Zhang et al., 2023) projects speech features into LLM input space for dialogue. LLM-CODEC aligns audio and text representations inside the hidden layers of a frozen LLM. This choice directly targets generation and token predictability.

Token consistency and factorized tokenizers. Recent work studies token instability under acoustic perturbations. Liu et al. (2025) identify the Discrete Representation Inconsistency (DRI) phenomenon, attributing it to encoder receptive field leakage across contexts, and propose consistency regularizers to mitigate it. Wu et al. (2025) propose codec-LM co-design techniques including a framewise encoder and codebook-level dropout to

improve end-to-end TTS performance. Other lines decouple semantic and acoustic factors, or use spectral quantization to obtain more predictable token sequences by construction. LLM-CODEC is complementary: it diagnoses a different root cause, the objective mismatch between reconstruction and prediction, and addresses it through LLM-facing training objectives without changing model architectures.

3 Preliminaries

3.1 Neural Audio Codecs

A neural audio codec compresses audio into discrete tokens. Let $\mathcal{E}$ be the encoder, $\mathcal{Q}$ the quantizer, and $\mathcal{D}$ the decoder. Given waveform $x \in \mathbb{R}^T$, the codec produces:

$$z = \mathcal{E}(x), \quad c = \mathcal{Q}(z), \quad \hat{x} = \mathcal{D}(c). \tag{1}$$

The training objective is reconstruction:

$$\mathcal{L}_{\text{codec}} = \mathcal{L}_{\text{recon}}(x, \hat{x}) + \mathcal{L}_{\text{VQ}}(z), \tag{2}$$

where $\mathcal{L}_{\text{recon}}$ includes time-domain L1, mel-spectrogram L1, and adversarial losses. $\mathcal{L}_{\text{VQ}}$ is the commitment loss for the quantizer. Modern codecs like EnCodec (Défossez et al., 2022) use Residual Vector Quantization (RVQ) with multiple codebooks. This achieves high reconstruction quality at low bitrates (e.g., 6 kbps).

3.2 Spoken Language Models

A spoken language model treats codec tokens as a language and models their distribution autoregressively:

$$p(c_1, \ldots, c_T) = \prod_{t=1}^{T} p(c_t | c_{<t}). \tag{3}$$

This formulation allows SLMs to leverage powerful transformer architectures developed for text LLMs. Systems like VALL-E (Wang et al., 2023), AudioLM (Borsos et al., 2023), and SpeechGPT (Zhang et al., 2023) have demonstrated impressive speech generation capabilities.

3.3 The Objective Mismatch Problem

The codec and the LLM are optimized for fundamentally different goals.

Codec goal: preserve all information needed for reconstruction, including linguistic content, speaker identity, prosody, and fine-grained acoustic details.

LLM goal: model sequential dependencies, which favors token sequences that exhibit predictable patterns given context.

These goals can conflict: to faithfully reconstruct acoustic nuances, a codec may assign different tokens to acoustically distinct but linguistically equivalent realizations. For example, the same word "hello" can map to different token sequences under changes in pitch, speaking rate, or background noise. From the LLM's perspective, such variation appears as noise and reduces token predictability.

A concrete example. Consider the word "hello" spoken twice by the same speaker. The two utterances are linguistically identical but acoustically slightly different (different pitch contour, duration, breath). A reconstruction-oriented codec might encode them as:

$$\text{"hello"}_1 \rightarrow [1042, 3891, 2847, 1923]$$
$$\text{"hello"}_2 \rightarrow [1042, 3892, 2851, 1919]$$

The tokens are similar but not identical. This creates two problems for the LLM. First, the model must learn multiple token patterns for the same word, increasing the effective vocabulary complexity. Second, during generation, even if the model correctly predicts "hello," it must choose among acoustically valid variants, introducing unnecessary degrees of freedom. These variations accumulate: a sentence has exponentially many valid token sequences, making both learning and generation harder.

We note that token variability under acoustic perturbation has been independently studied by Liu et al. (2025), who attribute it to context leakage in encoder receptive fields. Our analysis differs in both the input condition (natural acoustic variation vs. contextual truncation) and the diagnosed root cause (reconstruction objective mismatch vs. receptive field leakage). Both perspectives are complementary and identify the same symptom from different angles.

3.4 Desiderata for LLM-Friendly Tokens

Based on this analysis, we identify two properties that LLM-friendly tokens should have:

Property 1: Multi-step predictability. Given context, not just the next token but several future tokens should be predictable. Linguistic units (phonemes, words, phrases) span multiple tokens. If a word starts, the LLM should be able to predict how it ends.

Property 2: Semantic consistency. Tokens representing the same linguistic content should produce similar LLM representations, regardless of acoustic variations. "Hello" should look like "hello" inside the LLM, whether spoken loudly or softly.

Standard codecs satisfy neither property. They optimize reconstruction, not predictability or semantic consistency.

4 LLM-Codec

We propose to train the codec with LLM-facing objectives. Figure 1 shows the overall architecture. The key components are: (1) future token prediction heads, (2) semantic alignment losses, and (3) a differentiable Gumbel bridge.

4.1 Future Token Prediction (FTP)

Why multi-step prediction? Standard next-token prediction optimizes one-step dependencies. But linguistic structure spans multiple tokens. A phoneme at 50 Hz might be 2–4 tokens. A word might be 5–15 tokens. We want the codec to produce tokens where seeing the beginning of a word helps predict its ending.

Medusa-style heads. We add K prediction heads, where head M_k predicts the token at offset k. Each head is a linear projection:

$$M_k : \mathbb{R}^H \to \mathbb{R}^V, \quad M_k(h) = hW_k. \tag{4}$$

Initialization from LLM head. Let W_{LM} denote the frozen LLM output projection. Let $\mathcal{V}_{\text{audio}}$ be the index set of audio tokens in the extended vocabulary. We initialize each Medusa head by copying the audio-token sub-matrix from W_{LM}:

$$W_k \leftarrow \text{SelectAudio}(W_{\text{LM}}, \mathcal{V}_{\text{audio}}). \tag{5}$$

This initialization leverages the pretrained output geometry and stabilizes early training.

Inverse-distance weighting. Near-future predictability should dominate the signal, but farther horizons are still informative. We use inverse-distance weights and normalize them to sum to one:

$$w_k = \frac{1/k}{\sum_{j=1}^{K} 1/j}. \tag{6}$$

For $K = 5$, this yields $w \approx [0.44, 0.22, 0.15, 0.11, 0.09]$.

Loss formulation. Let T be the audio-token sequence length. We define the FTP loss as a weighted multi-step cross entropy:

$$\mathcal{L}_{\text{FTP}} = \frac{1}{T-K} \sum_{t=1}^{T-K} \sum_{k=1}^{K} w_k \cdot \text{CE}(M_k(h_t), c_{t+k}), \tag{7}$$

where h_t is the LLM hidden state at position t and c_{t+k} is the target token at offset k.

Gradient flow. Gradients from FTP flow through the LLM's hidden states, the input embeddings, the Gumbel bridge (which will be further introduced in Section 4.3), and finally to the codec encoder. This end-to-end gradient flow is what allows FTP to shape the codec's behavior.

4.2 Semantic Alignment (SA)

Why semantic alignment? FTP encourages predictability, but predictability alone is not enough. A codec could learn to produce predictable but semantically meaningless tokens. We need to ensure tokens preserve linguistic content.

Core idea. If audio and text represent the same content, they should produce similar representations inside the LLM. We enforce this by aligning hidden states from the audio branch with hidden states from the text branch.

Layer selection. Not all layers are equally suitable for alignment. Lower layers capture modality-specific surface features. Upper layers capture abstract semantics. Let L denote the total number of layers in the LLM. We align middle-to-high layers: $l \in [L/3, 0.8L]$. For a 32-layer LLM, this corresponds to layers 10–25.

Representation extraction. We align sequence-level states because speech and text have different token rates and lengths. For a causal transformer, the last hidden state summarizes the full prefix. For each selected layer l, we use last-position pooling:

- Audio: $h^{(l)}_{\text{audio}}$ is the hidden state at the last audio token.
- Text: $h^{(l)}_{\text{text}}$ is the hidden state at the last non-padding text token.

We compute the text branch with `no_grad` and detach $h^{(l)}_{\text{text}}$. This prevents the audio pathway from drifting the semantic geometry of text representations and stabilizes the alignment objective.

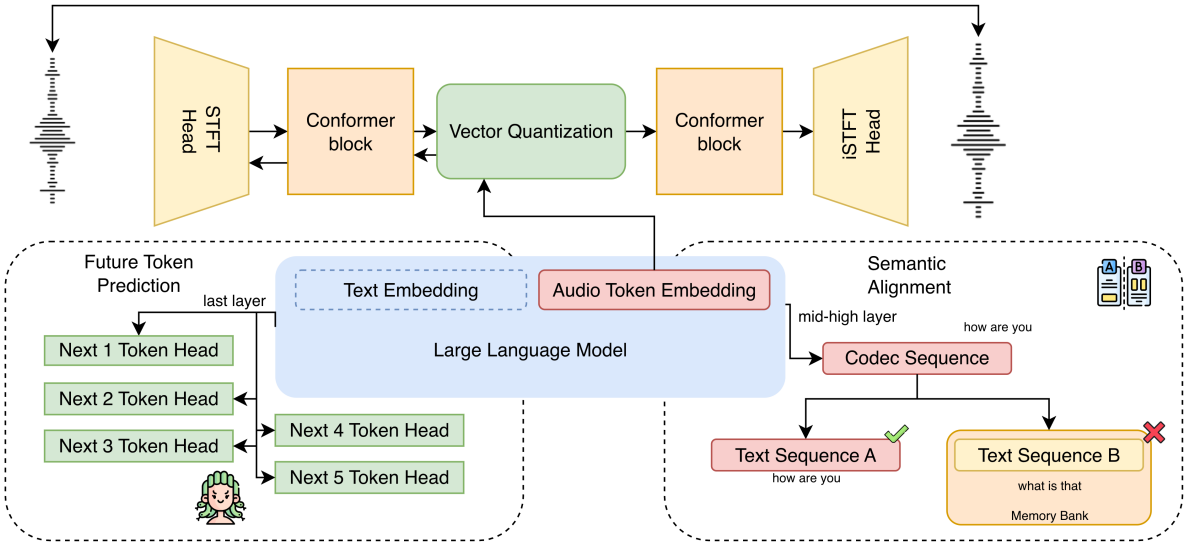

Figure 1: **Overview of LLM-CODEC.** Audio is encoded by the codec and passed through a Gumbel bridge to obtain differentiable embeddings. A single LLM forward pass produces hidden states for both FTP (using K Medusa heads) and SA (aligning with text representations). Gradients flow back through the bridge to update the codec encoder.

Cosine alignment loss. We minimize cosine distance across selected layers:

$$\mathcal{L}_{\text{cos}} = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \left(1 - \cos(\hat{h}^{(l)}_{\text{audio}}, \hat{h}^{(l)}_{\text{text}})\right), \tag{8}$$

where $\hat{h}$ denotes ℓ_2-normalized vectors.

Contrastive loss with memory bank. Cosine loss alone can cause representation collapse. We add contrastive learning to maintain discriminability. We maintain a memory bank Q of recent text representations (FIFO queue, size 512). For each audio representation, the paired text is positive, and bank entries are negatives:

$$\mathcal{L}_{\text{ctr}} = -\log \frac{\exp(\alpha \cdot \text{sim}(h_a, h_t^+))}{\exp(\alpha \cdot \text{sim}(h_a, h_t^+)) + \sum_{q \in Q} \exp(\alpha \cdot \text{sim}(h_a, q))} \tag{9}$$

where $\alpha = 5.0$ is the logit scale and we apply label smoothing with $\epsilon = 0.1$.

4.3 Differentiable Gumbel Bridge

Vector quantization uses argmax, which has zero gradient almost everywhere. We use the Gumbel-Softmax trick (Jang et al., 2017) with hard sampling: discrete tokens in the forward pass, smooth gradients in the backward pass. Let $z_t \in \mathbb{R}^C$ be the codec's continuous latent at time t. A linear projection maps it to logits $\ell_t = z_t W_{\text{bridge}}$, and we apply $y_t = \text{GumbelSoftmax}(\ell_t/\tau, \text{hard=True})$. The LLM embedding is $e_t = y_t E_{\text{audio}}$, where $E_{\text{audio}} \in \mathbb{R}^{V \times H}$ is the audio token embedding matrix. We anneal temperature τ from 1.0 to 0.3 over 20k steps. To prevent the bridge from diverging from the codec's quantizer, we add $\mathcal{L}_{\text{bridge}} = \text{CrossEntropy}(\ell_t, c_t)$, where c_t is the codec's original token.

4.4 Training Procedure

We activate LLM-facing objectives only after codec reconstruction stabilizes, using a delayed ramp-up schedule to avoid injecting high-variance gradients into early training. We stagger the two objectives so the codec first adapts to token predictability (FTP), then to semantic alignment (SA). FTP activates at step 10k and ramps to full weight by step 12k. SA activates at step 12k and ramps to full weight by step 14k. This staggering avoids gradient interference between the two objectives during initial ramp-up. The total loss combines the standard codec objective $\mathcal{L}_{\text{codec}}$, the bridge alignment $\mathcal{L}_{\text{bridge}}$, and the step-weighted LLM regularizers $\mathcal{L}_{\text{FTP}}$, $\mathcal{L}_{\text{cos}}$, and $\mathcal{L}_{\text{ctr}}$.

We optimize:

$$\begin{aligned}\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{codec}} + \lambda_{\text{bridge}}\mathcal{L}_{\text{bridge}} + \lambda_{\text{FTP}}\mathcal{L}_{\text{FTP}} \\ + \lambda_{\text{cos}}\mathcal{L}_{\text{cos}} + \lambda_{\text{ctr}}\mathcal{L}_{\text{ctr}}.\end{aligned} \tag{10}$$

We train the codec encoder and decoder, Gumbel bridge, audio token embeddings, and Medusa heads, while freezing the LLM backbone (to preserve text capability). We use a substantially lower learning rate for the decoder than for the auxiliary components, so that the reconstruction path-

way remains stable while the encoder adapts to LLM-facing signals. The active reconstruction loss further constrains the decoder to its pretrained operating region. Inference cost is unchanged because auxiliary heads are discarded at test time.

5 Experiments

5.1 Experiment Setup

Codec training. We train on LibriSpeech train-clean-100 with paired transcripts (Panayotov et al., 2015). We fine-tune the AUV (Chen et al., 2025) codec (both encoder and decoder) at 50 Hz (vocabulary size 20,480). We use Qwen3-4B-Instruct (Yang et al., 2025) (32 layers, hidden size 2,560) as a frozen LLM backbone to isolate the effect of tokenization. We train the codec for 25k steps with 4-second segments and an effective batch size of 10. Appendix A reports the full configuration and hyperparameters.

SLM training. We use the same LM training recipe across tokenizers. We tokenize LibriSpeech train-clean-100 with each codec. We fine-tune Qwen3-4B-Instruct (Yang et al., 2025) with LoRA (Hu et al., 2022) (rank 64, α=128, dropout 0.05, applied to all linear layers) using next-token prediction with learning rate 10^{-4} and effective batch size 32. We use a cosine learning rate scheduler with 10% warmup. For LLM-CODEC, we start from the codec-trained checkpoint which already contains the expanded audio vocabulary and trained audio-token embeddings. For baselines, we expand the vocabulary at the start of SLM training. In both cases, the LLM backbone is Qwen3-4B-Instruct, and LoRA adapts all linear layers identically. We train for 3 epochs and evaluate on SALMon (Maimon et al., 2025).

SLM evaluation. We measure token learnability by training a token-level speech LM and evaluating coherence on SALMon. SALMon evaluates whether a model assigns higher likelihood to coherent speech than to minimally perturbed incoherent variants. Each example contains a coherent sample and an incoherent counterpart constructed by changing one acoustic factor mid-utterance. We select the coherent sample by comparing length-normalized negative log-likelihood under the token LM. We report the speaker and acoustic environment categories in the main paper, since they most directly reflect token-level stability under mid-utterance perturbations. We report the emotion categories in the appendix.

Reconstruction evaluation. We evaluate codec reconstruction on Codec-SUPERB-tiny (Wu et al., 2024) across Speech, Music, and Environmental Audio. We report domain-appropriate metrics to avoid misleading comparisons. Main-text tables use Speech (Mel, STFT, PESQ, STOI), Music (Mel, STFT, PESQ, STOI, F0), and Audio (Mel, STFT, PESQ). Appendix B provides the full metric applicability rationale.

Baselines. We compare against strong codec tokenizers that are commonly used for speech or audio language modeling. We match the token rate to 50 tokens/s to control sequence length and modeling difficulty. The baselines include AUV (Chen et al., 2025) (our starting point), BigCodec (Xin et al., 2024), UniCodec (Jiang et al., 2025), and WavTokenizer (Ji et al., 2025) variants with different capacities.

5.2 Speech Language Modeling

Reconstruction fidelity is not sufficient for spoken language modeling. We therefore first evaluate whether LLM-CODEC produces more LLM-friendly token sequences by training token-level LMs and testing speech coherence on SALMon.

Main results. Table 1 shows SALMon accuracy. Three findings stand out.

(1) LLM-CODEC *outperforms all baselines by a large margin.* LLM-CODEC achieves 61.6% overall accuracy, compared to 49.4% for AUV (+12.1 points). All baselines cluster near chance level (48–50%), regardless of their reconstruction quality.

(2) Gains are broad across categories. LLM-CODEC improves over AUV on every category. The largest gains appear on background consistency (BG-All: 71.5% vs. 49.0%, +22.5 points) and reverberation (RIR: 62.5% vs. 44.0%, +18.5 points).

(3) Reconstruction quality does not predict learnability. UniCodec and BigCodec achieve competitive reconstruction scores (Section 5.3), yet their SALMon accuracy is no better than AUV or WavTokenizer. This confirms that the reconstruction-prediction mismatch, not codec capacity, is the bottleneck.

SALMon isolates a property that matters for token LMs, namely likelihood contrast between coherent and minimally perturbed incoherent audio. This likelihood-based protocol directly tests

Model	Speaker		Acoustic Environment				Overall
	Spkr	Gend	RIR	BG-Align	BG-Dom	BG-All	
WavTok-L	47.0	52.5	37.5	51.5	50.5	51.0	48.3
BigCodec	50.5	49.5	43.5	48.0	53.5	48.5	49.4
UniCodec	49.0	53.0	53.0	47.5	45.5	46.0	50.1
AUV	47.5	52.5	44.0	45.5	53.5	49.0	49.4
LLM-CODEC	**63.0**	**65.0**	**62.5**	**48.0**	**69.0**	**71.5**	**61.6**

Table 1: **SALMon speech coherence evaluation.** Accuracy (%) on speaker and acoustic consistency after 3 epochs of LM training. Overall is the 8-category SALMon average (including sentiment, reported in Appendix). All baselines cluster around chance level (48–50%). LLM-CODEC achieves 61.6% overall (+12.1 over AUV). Gains are largest on background consistency (BG-All: +22.5, BG-Dom: +15.5) and reverberation (RIR: +18.5).

whether a tokenizer produces sequences that an LM can model reliably.

Why do speaker-related scores improve? We hypothesize that SA encourages representations that are more invariant to speaker-specific nuisance factors given the same transcript. Such invariance can make mid-utterance speaker changes appear as sharper distribution shifts, which increases likelihood contrast on SALMon.

Initialization matters. We initialize audio token embeddings from the LLM's text embedding space. A random initialization reduces SALMon accuracy to 50.7%, which is close to chance for a binary-choice benchmark. This result suggests that reusing the LLM's pre-trained embedding geometry stabilizes early training for speech tokens.

Token predictability. Beyond SALMon accuracy, we directly measure token predictability via language modeling perplexity. Figure 2 shows validation perplexity on LibriSpeech for each codec. All baselines exhibit perplexity in the range 148K–160K, regardless of codec parameter count (80M–211M). In contrast, LLM-CODEC achieves perplexity of 4,617, which is 35× lower than AUV (159,768) despite identical parameter counts (122.63M). This result confirms that improvements stem from training objectives, not model capacity. The perplexity reduction directly validates our core claim: LLM-facing objectives produce more predictable tokens.

Implications. SALMon results support our central claim that LLM-facing objectives can shape a codec tokenizer toward more learnable token sequences. The improvement over AUV indicates that token predictability is a first-order bottleneck for spoken language modeling.

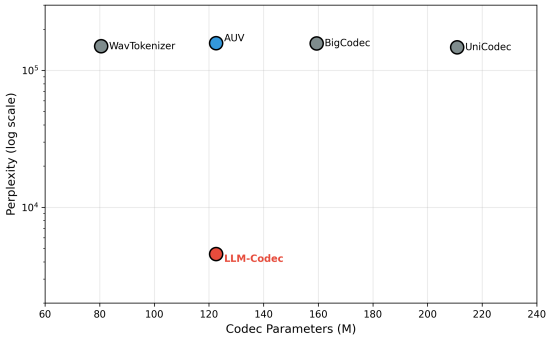

Figure 2: **Perplexity is determined by training objectives, not model size.** All baselines (80M–211M parameters) achieve similar perplexity (148K–160K). LLM-CODEC (122M, same as AUV) achieves 4,617, a 35× reduction. This confirms that the codec-LLM objective mismatch, not model capacity, is the bottleneck.

5.3 Reconstruction Quality

We next evaluate reconstruction quality to verify that LLM-facing objectives do not degrade codec fidelity. Table 2 summarizes reconstruction across domains.

Speech domain: LLM-CODEC improves spectral fidelity. On speech, LLM-CODEC achieves the best spectral fidelity with Mel distance 0.724 and STFT distance 1.599. Relative to AUV, LLM-CODEC improves Mel by 5.0% and STFT by 3.0%. Perceptual metrics also improve slightly: PESQ from 2.094 to 2.102 and STOI from 0.850 to 0.859. BigCodec achieves the highest perceptual scores (PESQ 2.208, STOI 0.877), but LLM-CODEC closes most of the gap while leading on spectral metrics. Combined with the 35× perplexity reduction (Section 5.2), this result demonstrates that LLM-facing training can improve both reconstruction fidelity and token learnability simultaneously.

Why does reconstruction improve? We train both the codec encoder and decoder at a low

Model	Speech				Music					Audio		
	Mel↓	STFT↓	PESQ↑	STOI↑	Mel↓	STFT↓	PESQ↑	STOI↑	F0↑	Mel↓	STFT↓	PESQ↑
BigCodec	0.810	1.718	**2.208**	**0.877**	1.522	3.108	1.922	0.606	0.728	2.101	4.864	1.391
UniCodec	0.830	1.824	2.022	0.851	1.196	**2.441**	1.859	0.588	0.760	**1.337**	**3.336**	1.548
WavTok-M	0.904	1.846	1.843	0.823	1.407	2.683	1.579	0.528	0.683	1.382	3.367	1.380
WavTok-L	1.133	2.006	1.547	0.765	1.558	2.896	1.484	0.478	0.722	1.579	3.777	1.334
WavTok-S	1.096	2.174	1.437	0.761	1.631	2.976	1.350	0.475	0.629	1.487	3.442	1.273
AUV (base)	0.762	1.648	2.094	0.850	1.129	2.557	**2.195**	0.609	**0.785**	1.847	4.407	**1.567**
LLM-CODEC	**0.724**	**1.599**	2.102	0.859	**1.126**	2.541	2.175	**0.614**	0.767	1.836	4.389	1.562

Table 2: **Reconstruction quality across domains.** Evaluated on Codec-SUPERB-tiny (Wu et al., 2024). All models operate at 50 tokens/s. We report domain-appropriate metrics: F0 only for Music (pitch matters for melody), STOI only for Speech/Music (intelligibility undefined for environmental sounds). **Speech**: LLM-CODEC achieves best spectral fidelity (Mel 0.724, STFT 1.599), improving over AUV by 5.0% and 3.0% respectively. **Music**: LLM-CODEC matches or slightly improves AUV. **Audio**: AUV leads on PESQ.

learning rate (5×10^{-6}) with multiple reconstruction losses (mel, multi-scale mel, multi-resolution STFT, complex STFT, and GAN). The active reconstruction losses constrain both encoder and decoder to remain within the pretrained operating region, while allowing gradual adaptation. Our ablation (Table 3) confirms that the reconstruction improvement comes from this training procedure, not from FTP or SA specifically.

Music domain: comparable to AUV. On music, LLM-CODEC matches or slightly improves AUV. Mel distance is marginally better (1.126 vs. 1.129), and STOI improves (0.614 vs. 0.609). AUV retains a small advantage on PESQ (2.195 vs. 2.175) and F0 correlation (0.785 vs. 0.767). The differences are small, confirming that our speech-centric training does not harm music reconstruction.

Audio (environmental) domain: comparable to AUV. On environmental audio, UniCodec achieves the best spectral metrics, while LLM-CODEC is comparable to AUV. LLM-CODEC slightly improves Mel (1.836 vs. 1.847) but slightly reduces PESQ (1.562 vs. 1.567).

Domain-specific summary. LLM-CODEC improves speech spectral fidelity by 5.0% Mel while preserving or slightly improving reconstruction on music and environmental audio. The improvements on speech are consistent with the training procedure, which uses speech-text supervision and GAN-based adversarial training. The learnability gains (35× perplexity, +12.1 SALMon) are orthogonal to and come on top of these reconstruction improvements.

5.4 Analysis

Learnability versus reconstruction. A key finding of this work is the dissociation between reconstruction quality and token learnability. All baselines achieve comparable reconstruction on speech (Table 2), yet their token-level LMs plateau at chance-level SALMon accuracy (Table 1). LLM-CODEC improves both reconstruction (5.0% speech Mel) and learnability (+12.1 SALMon, 35× perplexity). Importantly, our ablation shows that the reconstruction improvement comes from the training procedure (GAN, multi-scale losses), while the learnability improvement comes from FTP and SA. These two effects are additive and orthogonal.

What makes tokens learnable? Our results suggest that learnability requires two complementary properties, both introduced by LLM-CODEC. First, tokens should be *locally predictable*: given context, the next few tokens should be deterministic. FTP enforces this by penalizing unpredictable futures. Second, tokens should be *semantically grounded*: the same linguistic content should map to similar representations regardless of acoustic variation. SA enforces this by aligning audio and text branches.

5.5 Ablation Study

We ablate each objective on speech reconstruction to isolate the roles of FTP and SA.

Component ablation. Table 3 ablates FTP and SA on both reconstruction fidelity and token learnability. All variants improve Mel distance by ∼5% over the original AUV, but are indistinguishable from each other on reconstruction (≤0.002 Mel). This confirms that the reconstruction improvement comes from the shared training procedure (GAN, multi-scale losses), not from FTP or SA.

Variant	Mel↓	PPL↓	SALMon↑
AUV (original)	0.762	159,768	49.4
FTP only	0.725	4,631	61.8
SA only	0.723	4,616	61.3
LLM-CODEC (FTP + SA)	0.724	4,617	61.6

Table 3: **Component ablation (Speech).** All variants improve Mel ∼5% over AUV (from training procedure). All variants achieve ∼35× lower PPL and ∼+12 SALMon (from LLM-facing objectives). FTP and SA each independently capture the full learnability gain.

K	Mel↓	PPL↓	SALMon↑
AUV (original)	0.762	159,768	49.4
$K = 1$	0.725	4,561	61.6
$K = 3$	0.723	4,554	61.4
$K = 5$ (default)	0.724	4,617	61.6
$K = 10$	0.724	4,539	61.4

Table 4: **Effect of prediction horizon K (Speech).** All metrics are invariant to K. Even K=1 achieves the full learnability gain, suggesting that the key factor is the presence of LLM-facing gradients, not the prediction horizon length.

On learnability, all LLM-Codec variants achieve dramatically lower perplexity (4.6K vs. 160K) and higher SALMon accuracy (61–62% vs. 49%). FTP-only and SA-only each achieve similar learnability to the full model, suggesting that each objective independently captures most of the benefit.

Effect of prediction horizon K. Table 4 ablates the number of Medusa heads K in FTP. Reconstruction, perplexity, and SALMon are all invariant to K. This indicates that even K=1 (next-token prediction only) suffices to reshape the token space for learnability, and multi-step prediction does not provide additional gains in our setting.

6 Discussion

Why does FTP help? FTP directly regularizes token sequences to be predictable beyond the next step. Our ablation confirms that FTP alone achieves most of the learnability gain (SALMon 61.8 vs. 49.4 for AUV). Interestingly, even K=1 (single-step prediction) suffices, suggesting that the key mechanism is the LLM-facing gradient signal itself, not the multi-step horizon.

Why does SA help? SA injects an external semantic anchor by tying audio-induced hidden states to text-induced hidden states. SA alone also achieves strong learnability (SALMon 61.3), confirming that semantic grounding independently improves token predictability. FTP and SA target different failure modes, since FTP emphasizes local predictability and SA emphasizes semantic invariance. In practice, the two objectives achieve similar gains independently, and their combination does not yield additive improvement.

Reconstruction improvement is a byproduct. Our ablation reveals that the 5.0% speech Mel improvement over AUV comes from the training procedure (GAN, multi-scale losses, low-LR fine-tuning), not from FTP or SA. All ablation variants achieve identical reconstruction. This means the learnability gains (35× perplexity, +12.1 SALMon) come entirely from LLM-facing objectives, while the reconstruction gains come entirely from the training procedure. The two effects are independent and additive.

Domain effects. The SALMon improvements are speech-specific because SA relies on speech-text correspondence. For non-speech audio, SA becomes ill-posed or indirect. This gap motivates future work on alternative supervision for music and environmental sound.

Computation. Training requires an extra frozen-LLM forward pass, a GAN discriminator update, and auxiliary prediction heads. The total training budget is 25k steps. Inference is unchanged, since the deployed codec keeps the same encoder-decoder structure and discards the auxiliary heads.

7 Conclusion

We proposed LLM-CODEC, which trains a neural audio codec with objectives that reflect downstream autoregressive modeling. LLM-CODEC adds future token prediction and semantic alignment through a differentiable bridge, without modifying the codec or LLM architectures. Experiments show that LLM-CODEC improves both reconstruction fidelity (5.0% speech Mel) and token learnability (SALMon +12.1 points, perplexity 35× lower). Ablation confirms that these gains are additive: reconstruction improves from the training procedure, while learnability improves from FTP and SA. These results indicate that tokenizer predictability, not reconstruction quality, is the key bottleneck for spoken language modeling.

Limitations

Speech-centric supervision. SA relies on speech-text correspondence, so it requires paired transcripts during training. This assumption holds for read-speech corpora such as LibriSpeech, but it is weaker for untranscribed audio or non-speech domains. Our cross-domain results reflect this limitation and motivate alternative alignment signals beyond text.

Frozen LLM backbone. We freeze the LLM backbone to preserve its text competence and to isolate the effect of tokenizer training. Jointly adapting the LLM could further improve speech modeling, but it may introduce regressions on text tasks and complicate attribution.

Evaluation coverage. Our primary evaluations emphasize read speech. Conversational settings contain disfluencies, overlap, and rapid speaker turns, which may stress different token properties. Future work should validate LLM-CODEC under conversational corpora and multi-speaker conditions.

Training overhead. The auxiliary Medusa heads increase training-time memory and compute, even though they are discarded for inference. This cost may limit scaling to larger backbones or larger batches.

Learnability ablation. Our ablation study confirms that reconstruction is invariant to the choice of LLM-facing objectives and prediction horizon K. However, we do not ablate the learnability impact (SALMon, perplexity) of each component separately. Future work should measure how FTP-only, SA-only, and different K values affect downstream speech coherence.

Acknowledgement

This work was supported by the Ministry of Education (MOE) of Taiwan under the project Taiwan Centers of Excellence in Artificial Intelligence, through the NTU Artificial Intelligence Center of Research Excellence (NTU AI-CoRE). We thank the National Center for High-performance Computing (NCHC) of the National Applied Research Laboratories (NARLabs) in Taiwan for providing computational and storage resources.

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. 2023. Audiolm: A language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2523–2533.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. MEDUSA: Simple LLM inference acceleration framework with multiple decoding heads. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 5209–5235.

Yushen Chen, Kai Hu, Long Zhou, Shulin Feng, Xusheng Yang, Hangting Chen, and Xie Chen. 2025. Auv: Teaching audio universal vector quantization with single nested codebook. *arXiv preprint arXiv:2509.21968*.

Ju-Chieh Chou, Jiawei Zhou, and Karen Livescu. 2025. Flow-SLM: Joint learning of linguistic and acoustic information for spoken language modeling. *arXiv preprint*. ArXiv:2508.09350. ASRU 2025.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2022. High fidelity neural audio compression. *Preprint*, arXiv:2210.13438. TMLR 2023 (Featured Certification, Reproducibility Certification). OpenReview: ivCd8z8zR2.

Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. 2023. CLAP: Learning audio concepts from natural language supervision. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *arXiv preprint*. ArXiv:2106.07447.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. volume 1, page 3.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*.

Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, Zehan Wang, Ruiqi Li, Ziang Zhang, Xiaoda Yang, Rongjie Huang, Yidi Jiang, Qian Chen, Siqi Zheng, and Zhou Zhao. 2025. Wavtokenizer: An efficient acoustic discrete codec tokenizer for audio language modeling. In *International Conference on Learning Representations*.

Yidi Jiang, Qian Chen, Shengpeng Ji, Yu Xi, Wen Wang, Chong Zhang, Xianghu Yue, ShiLiang Zhang, and Haizhou Li. 2025. Unicodec: Unified audio codec with single domain-adaptive codebook. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19112–19124.

Wenrui Liu, Zhifang Guo, Jin Xu, Yuanjun Lv, Yunfei Chu, Zemin Liu, and Junyang Lin. 2025. Analyzing and mitigating inconsistency in discrete speech tokens for neural codec language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31035–31046, Vienna, Austria. Association for Computational Linguistics.

Gallil Maimon, Amit Roth, and Yossi Adi. 2025. Salmon: A suite for acoustic language model evaluation. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. ArXiv:2409.07437.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. 2023. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint*. ArXiv:2301.02111.

Haibin Wu, Ho-Lam Chung, Yi-Cheng Lin, Yuan-Kuei Wu, Xuanjun Chen, Yu-Chi Pai, Hsiu-Hsuan Wang, Kai-Wei Chang, Alex Liu, and Hung-yi Lee. 2024. Codec-superb: An in-depth analysis of sound codec models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10330–10348.

Shih-Lun Wu, Aakash Lahoti, Arjun D Desai, Karan Goel, Chris Donahue, and Albert Gu. 2025. Towards codec-LM co-design for neural codec language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 4: Student Research Workshop)*, pages 55–65, Albuquerque, USA. Association for Computational Linguistics.

Detai Xin, Xu Tan, Shinnosuke Takamichi, and Hiroshi Saruwatari. 2024. Bigcodec: Pushing the limits of low-bitrate neural speech codec. *Preprint*, arXiv:2409.05377.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. SoundStream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507. ArXiv:2107.03312.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. 2023. SpeechGPT: Empowering large language models with intrinsic cross-modal conversational abilities. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15757–15773.

Xin Zhang, Dong Zhang, Shimin Li, Yaqian Zhou, and Xipeng Qiu. 2024. SpeechTokenizer: Unified speech tokenizer for speech large language models. In *International Conference on Learning Representations*. ArXiv:2308.16692.

A Implementation Details

This appendix reports the full implementation details of LLM-CODEC. This appendix specifies model configurations, training hyperparameters, and stability settings. This appendix also clarifies how we compute and report reconstruction metrics across domains.

A.1 Model Configurations

Codec. We use the AUV (Chen et al., 2025) codec at a 50 Hz token rate. The codec uses a vocabulary size of 20,480. The codec latent dimension is 256. The codec hop length is 320. We fine-tune both the codec encoder and decoder at a low learning rate (5×10^{-6}) to allow gradual adaptation while preserving reconstruction quality.

LLM backbone. We use Qwen3-4B-Instruct (Yang et al., 2025) as the frozen LLM backbone. The LLM has 32 transformer layers. The LLM hidden dimension is 2,560. We extend the LLM vocabulary with 20,480 audio tokens via `resize_token_embeddings`. We train only the audio-token embeddings. We freeze all other LLM parameters.

Gumbel bridge. We implement the Gumbel bridge as a single linear projection from codec latents to audio-token logits. We use `nn.Linear`$(256, 20480)$. We anneal the Gumbel-Softmax temperature from 1.0 to 0.3 over 20k steps. We use a cosine schedule for annealing.

Medusa heads (FTP). We use $K = 5$ Medusa-style prediction heads. Each head is a bias-free linear layer: `nn.Linear`$(2560, 20480,$ `bias=False`$)$. We initialize each head from the frozen LLM output projection restricted to audio token indices. We implement this initialization as `lm_head.weight[audio_ids]`.

A.2 Training Configuration

Table 5 lists the main training hyperparameters. We use 4-second audio segments. We use gradient accumulation to reach an effective batch size of 10. We train for 25k optimization steps. We clip the gradient norm to 15.0. We use SGD with momentum 0.9 for the codec encoder and decoder, and AdamW for the audio token embeddings and Medusa heads.

Hyperparameter	Value
Batch size	1
Gradient accumulation	10
Effective batch size	10
Segment length	4 seconds
LR (encoder)	5×10^{-6}
LR (decoder)	5×10^{-6}
LR (embeddings + heads)	1×10^{-4}
Codec optimizer	SGD (momentum=0.9, wd=1e-4)
Embed optimizer	AdamW (β_1=0.9, β_2=0.99)
Gradient clip	15.0
Total steps	25k
Warmup	2k steps
λ_{mel}	1.5
$\lambda_{\text{ms-mel}}$	0.5
$\lambda_{\text{mr-stft}}$	0.5
λ_{cstft}	0.8 (phase weight 0.5)
λ_{FTP}	0.2 (after ramp)
λ_{cos}	0.1 (after ramp)
λ_{ctr}	0.05 (after ramp)
D-only warmup	0–10k steps
FTP schedule	delay 10k, warmup 2k
SA schedule	delay 12k, warmup 2k

Table 5: Training hyperparameters.

A.3 Semantic Alignment Details

Table 6 lists the semantic alignment hyperparameters. We align a mid-to-high layer range to target semantic representations. We use a memory bank to provide negatives for the contrastive loss. We use a fixed logit scale and label smoothing.

A.4 Training Phases

Training proceeds in three phases over 25k steps.

Phase 1: D-only warmup (steps 0–10k). Only the GAN discriminators are updated. The codec encoder and decoder remain in train mode (so VQ EMA statistics continue tracking), but their optimizer steps are skipped. Audio token embeddings and Medusa heads are updated throughout all phases.

Parameter	Value
Layer range	$[L/3, 0.8L] = [10, 25]$
Layer weights	Uniform
Memory bank size	512
Logit scale α	5.0
Label smoothing ϵ	0.1

Table 6: Semantic alignment hyperparameters.

Phase 2: Full training + FTP (steps 10k–12k). The codec encoder and decoder optimizers activate. FTP loss ramps from zero to full weight over 2k steps. SA remains inactive.

Phase 3: Full training + FTP + SA (steps 12k–25k). SA cosine and contrastive losses ramp from zero to full weight over 2k steps. All objectives are active for the remaining 11k steps.

This staggered schedule ensures the codec first learns basic reconstruction (Phase 1), then adapts to token predictability (Phase 2), and finally incorporates semantic grounding (Phase 3).

A.5 Reconstruction Losses

We train the codec with multiple reconstruction losses. We use a log mel-spectrogram ℓ_1 loss with 100 mel bins and hop size 256, with per-sample RMS normalization applied to both input and reconstruction before computing the loss. We use multi-scale mel losses at FFT sizes 512, 1024, and 2048 with 80 mel bins. We use a multi-resolution STFT loss combining spectral convergence and log magnitude error. We use a complex STFT loss that additionally measures phase distance via unit complex vector difference (phase weight 0.5), computed at FFT sizes 512, 1024, and 2048. We omit STOI and PESQ from the training loss. We report STOI and PESQ only for evaluation.

Phase jitter. We apply random phase jitter (± 24 samples) exclusively to the LM branch input. This augments the LM pathway with slight temporal perturbations while keeping the reconstruction pathway unaffected, encouraging the encoder to produce tokens that are stable under small phase shifts.

A.6 GAN Training

We enable adversarial training with a discriminator-only warmup phase. The discriminator trains alone

for the first 10k steps, allowing it to develop meaningful feature maps before providing gradients to the generator. We use a multi-period discriminator (MPD) with periods $\{2, 3, 5, 7, 11\}$. We use a multi-scale discriminator (MSD) with three scales and AvgPool downsampling. We use hinge loss with feature matching (initial weight 1.5, decaying to 1.0). We pause GAN updates if the feature-matching loss exceeds 99% of the total loss, for 500 steps when this condition holds. We apply R1 regularization with $\gamma = 2.0$ every 16 steps. GAN discriminators run in FP32 precision for stability.

A.7 Numerical Stability

We apply several stability mechanisms. We skip parameter updates when the loss is non-finite. We clamp logits to $[-80, 80]$ before softmax. We clip audio samples to $[-1.2, 1.2]$. We apply code noise by randomly replacing 1.5% of tokens as regularization for the LM branch. We set all BatchNorm layers in the codec to eval mode throughout training to prevent batch statistics from drifting.

B Additional Results

B.1 Metric Applicability and Reporting Rationale

This section clarifies which reconstruction metrics are meaningful for each domain in Codec-SUPERB-tiny. Each metric assumes specific signal properties. We report a metric only when its assumptions hold. This protocol avoids misleading comparisons across domains.

Mel distance (Mel). Mel distance measures the ℓ_1 error between log mel-spectrograms. Mel distance is defined for all audio signals. Mel distance reflects coarse spectral fidelity. We report Mel for Speech, Music, and Environmental Audio.

STFT distance (STFT). STFT distance measures spectral reconstruction error in the short-time Fourier domain. We use a multi-resolution STFT variant with spectral convergence and magnitude error. STFT distance is defined for all audio signals. STFT distance captures finer spectral structure than mel features. We report STFT for Speech, Music, and Environmental Audio.

Perceptual Evaluation of Speech Quality (PESQ). PESQ is calibrated for human speech perception. PESQ assumes a speech-like signal. PESQ is therefore most meaningful for Speech and singing voice. We report PESQ for Speech and Music. We report PESQ for Environmental Audio for completeness. We interpret PESQ on Environmental Audio cautiously.

Short-Time Objective Intelligibility (STOI). STOI measures speech intelligibility using temporal envelopes in short-time bands. STOI assumes linguistic content. STOI is meaningful for Speech and singing voice. STOI is not meaningful for Environmental Audio. We report STOI for Speech and Music. We do not report STOI for Environmental Audio.

Fundamental frequency correlation (F0 Corr). F0 correlation measures pitch preservation by correlating estimated fundamental frequency trajectories. F0 correlation assumes a stable harmonic structure with a well-defined fundamental frequency. Many environmental sounds are non-harmonic. Many speech segments contain long unvoiced regions. F0 correlation is therefore most meaningful for Music. We report F0 Corr for Music only.

Summary of reported metrics. We use the following metric sets:

- Speech: Mel, STFT, PESQ, STOI.
- Music: Mel, STFT, PESQ, STOI, F0 Corr.
- Environmental Audio: Mel, STFT, PESQ.

B.2 SALMon Emotion Categories

Table 7 reports the emotion-related SALMon categories omitted from the main text.

Model	Sent-Align	Sent-Cons
WavTok-L	48.0	48.0
BigCodec	53.0	48.5
UniCodec	52.5	54.5
AUV	47.5	56.0
LLM-CODEC	49.5	**64.0**

Table 7: **SALMon emotion categories.** Sentiment consistency shows a clear gain (+8.0 over AUV). Sentiment alignment is comparable to baselines (49.5%).

B.3 Perplexity Comparison

Table 8 reports token-level perplexity for all codecs after 3 epochs of LM training on LibriSpeech train-clean-100.

B.4 Domain-Specific Detailed Results

This section reports full reconstruction results for each domain. Table 9 averages metrics over all

Model	Eval Loss	Perplexity
WavTok-L	11.91	148,122
UniCodec	11.92	150,197
BigCodec	11.96	156,448
AUV	11.98	159,768
LLM-CODEC	**8.44**	**4,617**

Table 8: **Token-level perplexity.** All baselines cluster in the 148K–160K range. LLM-CODEC achieves 4,617, a 35× reduction over AUV.

domains. Tables 10, 11, and 12 report domain-specific results.

Model	Mel↓	STFT↓	PESQ↑	STOI↑	F0↑
UniCodec	**1.121**	**2.534**	1.810	0.651	0.620
WavTok-M	1.231	2.632	1.600	0.602	0.587
WavTok-S	1.405	2.864	1.353	0.541	0.535
WavTok-L	1.423	2.893	1.455	0.549	0.585
BigCodec	1.478	3.230	1.840	0.660	0.635
AUV (base)	1.246	2.871	**1.952**	**0.660**	**0.650**
LLM-CODEC	1.229	2.843	1.946	0.664	0.630

Table 9: **Overall reconstruction quality** (6,000 samples across all domains). All models operate at 50 tokens/s. UniCodec achieves best spectral fidelity (Mel, STFT). LLM-CODEC improves over AUV on spectral metrics while remaining competitive on perceptual metrics.

Model	Mel↓	STFT↓	PESQ↑	STOI↑
BigCodec	0.810	1.718	**2.208**	**0.877**
UniCodec	0.830	1.824	2.022	0.851
WavTok-M	0.904	1.846	1.843	0.823
WavTok-S	1.096	2.174	1.437	0.761
WavTok-L	1.133	2.006	1.547	0.765
AUV (base)	0.762	1.648	2.094	0.850
LLM-CODEC	**0.724**	**1.599**	2.102	0.859

Table 10: **Speech domain results** (2,000 samples from 10 datasets). We omit F0 correlation because pitch accuracy is less central for speech than intelligibility. LLM-CODEC achieves best spectral fidelity (Mel 0.724, STFT 1.599), improving 5.0% over AUV. BigCodec leads on perceptual metrics (PESQ 2.208, STOI 0.877).

B.5 Ablation Study: Cross-Domain Results

The main text reports speech-only ablations (Table 3). This section reports cross-domain ablations for completeness. Table 13 reports Mel distance across Speech, Music, and Environmental Audio.

B.6 Evaluation Datasets

Table 14 lists the datasets in Codec-SUPERB-tiny (Wu et al., 2024). We uniformly sample 2,000 clips per domain. This protocol yields 6,000 total evaluation samples.

Model	Mel↓	STFT↓	PESQ↑	STOI↑	F0↑
UniCodec	1.196	**2.441**	1.859	0.588	0.760
WavTok-M	1.407	2.683	1.579	0.528	0.683
BigCodec	1.522	3.108	1.922	0.606	0.728
WavTok-L	1.558	2.896	1.484	0.478	0.722
WavTok-S	1.631	2.976	1.350	0.475	0.629
AUV (base)	1.129	2.557	**2.195**	0.609	**0.785**
LLM-CODEC	**1.126**	2.541	2.175	**0.614**	0.767

Table 11: **Music domain results** (2,000 samples from 6 datasets). AUV leads on PESQ (2.195) and F0 (0.785). LLM-CODEC slightly improves Mel (1.126 vs. 1.129) and STOI (0.614 vs. 0.609). UniCodec achieves the lowest STFT distance (2.441).

Model	Mel↓	STFT↓	PESQ↑
UniCodec	**1.337**	**3.336**	1.548
WavTok-M	1.382	3.367	1.380
WavTok-S	1.487	3.442	1.273
WavTok-L	1.579	3.777	1.334
BigCodec	2.101	4.864	1.391
AUV (base)	1.847	4.407	**1.567**
LLM-CODEC	1.836	4.389	1.562

Table 12: **Environmental audio results** (2,000 samples from 4 datasets). We omit STOI and F0 correlation because intelligibility and pitch are undefined for environmental sounds. LLM-CODEC slightly improves over AUV on Mel (1.836 vs. 1.847). UniCodec achieves the best spectral metrics.

B.7 Model Nomenclature

We use short names for readability in tables. We list the corresponding model identifiers below.

- **WavTok-L**: `wavtokenizer_24k_large_600_4096`
- **WavTok-M**: `wavtokenizer_24k_medium_600_4096`
- **WavTok-S**: `wavtokenizer_24k_small_600_4096`

B.8 Qualitative Examples

We provide audio samples in the supplementary material. The supplementary material includes:

- Reconstruction comparisons across codecs.
- Generation samples for baseline versus LLM-CODEC tokenizers.
- Robustness examples under noise and perturbations.

Variant	Speech Mel↓	Music Mel↓	Audio Mel↓
AUV (original)	0.762	1.129	1.847
FTP only	0.725	1.126	1.836
SA only	0.723	1.126	1.837
LLM-CODEC (FTP + SA)	0.724	1.126	1.836

Table 13: **Cross-domain ablation (Mel distance).** All variants improve $\sim$5% over the original AUV on Speech, but are indistinguishable from each other ($\leq$0.002). The improvement comes from the shared training procedure.

Dataset	Features
Speech (10 datasets, 2,000 samples)	
LibriSpeech	diverse speaker, read audiobooks
VoxCeleb1	diverse speaker, celebrities on YouTube
Speech Commands v1	spoken keyword commands
QUESST	multi-lingual, low resource language
VoxLingua107 Top 10	multi-lingual, YouTube content
Audio SNIPS	spoken commands, crowdsourced
IEMOCAP	affective speech
CREMA-D	affective speech
Libri2Mix	multi-speaker scenarios
LibriCount	multi-speaker scenarios
Environmental Audio (4 datasets, 2,000 samples)	
ESC-50	diverse audio source
FSD-50K	diverse audio source
Gunshot Triangulation	diverse audio source
Vocal Imitations	human imitation of sound
Music (6 datasets, 2,000 samples)	
OpenSinger	singing voice, Chinese song
M4Singer	singing voice, Chinese song
VocalSet	singing skill
NSynth	instrument notes
GTZAN Genre	diverse music genre
GTZAN Music Speech	instrument note

Table 14: Datasets in Codec-SUPERB-tiny. We uniformly sample 2,000 clips per domain for balanced evaluation.